



日本国特許庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日

Date of Application:

2001年 7月25日

出願番号

Application Number:

特願2001-224250

出願人

Applicant(s):

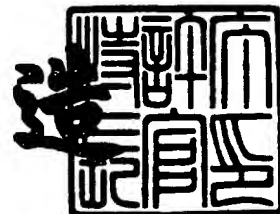
株式会社ソニー・コンピュータエンタテインメント

CERTIFIED COPY OF  
PRIORITY DOCUMENT

2001年 8月31日

特許庁長官  
Commissioner,  
Japan Patent Office

及川耕造



【書類名】 特許願

【整理番号】 SCEI01063

【提出日】 平成13年 7月25日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 11/00  
G06F 9/06

【発明の名称】 情報処理システム、実行可能モジュール生成方法および  
記憶媒体

【請求項の数】 24

【発明者】

【住所又は居所】 東京都中野区中央一丁目3 8 番1 号 住友中野坂上ビル  
株式会社シュガーアンドロケッツ内

【氏名】 山本 浩

【発明者】

【住所又は居所】 東京都中野区中央一丁目3 8 番1 号 住友中野坂上ビル  
株式会社シュガーアンドロケッツ内

【氏名】 大平 俊充

【特許出願人】

【識別番号】 395015319

【氏名又は名称】 株式会社 ソニー・コンピュータエンタテインメント

【代理人】

【識別番号】 100084032

【弁理士】

【氏名又は名称】 三品 岩男

【電話番号】 045(316)3711

【選任した代理人】

【識別番号】 100087170

【弁理士】

【氏名又は名称】 富田 和子

【電話番号】 045(316)3711

【先の出願に基づく優先権主張】

【出願番号】 特願2000-246404

【出願日】 平成12年 8月15日

【手数料の表示】

【予納台帳番号】 011992

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9912211

【プルーフの要否】 要

【書類名】明細書

【発明の名称】情報処理システム、実行可能モジュール生成方法および記憶媒体

【特許請求の範囲】

【請求項 1】

実行可能モジュールに含まれたプロテクトコードの矛盾により処理を終了させる手続きを含む、暗号化済みのプロテクト対象オブジェクトを記憶した記憶手段と、

前記記憶手段から前記暗号化済みプロテクト対象オブジェクトを読み出し、当該暗号化済みプロテクト対象オブジェクトを復号する復号手段と、

前記復号後のプロテクト対象オブジェクトと他のオブジェクトとの結合によって作成される実行可能モジュールに前記プロテクトコードを含ませるコード書込み手段と、

前記復号後のプロテクト対象オブジェクトを、前記他のオブジェクトとの結合を行った後に消去する削除手段と、

を有することを特徴とする情報処理装置。

【請求項 2】

実行可能モジュールに含まれた複数のプロテクトコードの矛盾により処理を終了させる手続きを含む、暗号化済みのプロテクト対象オブジェクトを記憶した記憶手段と、

前記記憶手段から前記暗号化済みプロテクト対象オブジェクトを読み出し、当該暗号化済みプロテクト対象オブジェクトを復号する復号手段と、

互いに関連付けられた第一のプロテクトコードと第二のプロテクトコードとを生成するコード生成手段と、

前記復号後のプロテクト対象オブジェクトに前記第一のプロテクトコードを埋め込み、当該第一のプロテクトコード埋込み済みのプログラム対象オブジェクトと他のオブジェクトとの結合により実行可能モジュールが作成されたら、当該実行可能モジュールに、前記第二のプロテクトコードを埋め込むコード書込み手段と、

前記第一のプロテクトコード埋込み済みのプロテクト対象オブジェクトを、前

記二のプロテクトコード埋込み前に消去する削除手段とを有することを特徴とする情報処理装置。

【請求項 3】

請求項 2 記載の情報処理装置であって、

前記コード生成手段は、

乱数から、前記第一のプロテクトコードと前記第二のプロテクトコードを生成することを特徴とする情報処理装置。

【請求項 4】

請求項 2 または 3 記載の情報処理装置であって、

前記コード書込み手段は、

前記第一のプロテクトコードと前記第二のプロテクトコードとにダミーデータを付加することを特徴とする情報処理装置。

【請求項 5】

請求項 1、2、3 および 4 のいずれか 1 項に記載の情報処理装置であって、

前記コード書込み手段は、前記実行可能モジュールに含ませるためのプロテクトコードを暗号化し、

前記プロテクト対象オブジェクトには、前記実行可能モジュールを読み込んだ装置に、当該実行可能モジュールに含まれた暗号化済みのプロテクトコードを、当該プロテクトコードの診断に際して復号させるための手続きを含む、

ことを特徴とする情報処理装置。

【請求項 6】

情報処理装置に処理を実行させるためのプログラムが格納された、機械読み取り可能な記憶媒体であって、

前記プログラムは、前記情報処理装置に、

実行可能モジュールに含まれたプロテクトコードの矛盾により処理を終了させる手続きを含むプロテクト対象オブジェクトを、暗号化済みのプロテクト対象オブジェクトの復号により生成する復号処理と、

前記復号処理で生成されたプロテクト対象オブジェクトと他のオブジェクトとの結合により、前記実行可能モジュールを作成する結合処理と、

前記結合処理による作成される実行可能モジュールに前記プロテクトコードを含ませるためのコード書込み処理と、

前記復号処理で生成されたプロテクト対象オブジェクトを、前記他のオブジェクトとの結合後に消去する削除処理と、

を実行させることを特徴とする記憶媒体。

【請求項 7】

情報処理装置に処理を実行させるためのプログラムが格納された、機械読み取り可能な記憶媒体であって、

前記プログラムは、前記情報処理装置に、

実行可能モジュールに含まれた複数のプロテクトコードの矛盾により処理を終了させる手続きを含むプロテクト対象オブジェクトを、暗号化済みのプロテクト対象オブジェクトの復号により生成する復号処理と、

互いに関連付けられた第一のプロテクトコードと第二のプロテクトコードとを生成するコード生成処理と、

前記復号処理後、当該復号処理で生成されたプロテクト対象オブジェクトに、前記第一のプロテクトコードを埋め込む第一コード書込み処理と、

前記コード書込み処理後、当該コード書込み処理で前記第一のプロテクトコードが埋め込まれたプロテクト対象オブジェクトと、他のオブジェクトとの結合により、実行可能モジュールを作成する結合処理と、

前記結合処理後、当該結合処理で作成された前記実行可能モジュールに前記第二プロテクトコードを埋め込む第二コード書込み処理と、

前記第一コード書込み処理後、前記第二コード書込み処理前に、前記復号処理で生成された前記プロテクト対象オブジェクトを消去する削除処理と、

を実行させることを特徴とする記憶媒体。

【請求項 8】

請求項 7 記載の記憶媒体であって、

前記プログラムは、前記コード生成処理において、前記情報処理装置に、

乱数から、前記第一のプロテクトコードと前記第二のプロテクトコードを生成させることを特徴とする記憶媒体。

【請求項 9】

請求項 7 または 8 記載の記憶媒体であって、  
前記プログラムは、前記情報処理装置に、  
前記第一のプロテクトコードと前記第二のプロテクトコードとにダミーデータを付加させることを特徴とする記憶媒体。

【請求項 1 0】

請求項 6、7、8 および 9 のいずれか 1 項に記載の記憶媒体であって、  
前記プログラムは、前記実行可能モジュールに含ませるためのプロテクトコードを暗号化する処理を前記情報処理装置に実行させ、  
前記プロテクト対象オブジェクトには、前記実行可能モジュールを読み込んだ装置に、当該実行可能モジュールに含まれた暗号化プロテクトコードを、当該プロテクトコードの診断に際して復号させるための手続きが含まれることを特徴とする記憶媒体。

【請求項 1 1】

情報処理装置によって処理されるオブジェクトを記憶した、機械読み取り可能な記憶媒体であって、  
暗号化されたプロテクト対象オブジェクトが格納され、  
当該プロテクト対象オブジェクトは、自身を組み込んだ実行可能モジュールに含まれた 1 以上のプロテクトコードに矛盾があれば処理を終了させる手続きを含む、  
ことを特徴とする記憶媒体。

【請求項 1 2】

請求項 1 1 記載の記憶媒体であって、  
前記プロテクト対象オブジェクトは、  
前記実行可能モジュールに含まれているプロテクトコードが暗号化されている場合に、当該プロテクトコードの診断に先立ち、当該プロテクトコードを復号させるための手続きを含む、  
ことを特徴とする記憶媒体。

【請求項 1 3】

複数のオブジェクトを結合して、実行可能モジュールを生成する処理を情報処理装置に実行させる実行可能モジュール生成方法であって、

実行可能モジュールに含まれたプロテクトコードの矛盾により処理を終了させる手続きを含むプロテクト対象オブジェクトを、暗号化済みのプロテクト対象オブジェクトの復号により生成し、

前記復号後のプロテクト対象オブジェクトと他のオブジェクトとの結合と、前記プロテクトコードの書込みとによって、前記実行可能モジュールを生成し、

前記復号後のプロテクト対象オブジェクトは、前記他のオブジェクトとの結合がなされた後に消去される、

ことを特徴とする実行モジュール生成方法。

【請求項 1 4】

複数のオブジェクトを結合して、実行可能モジュールを生成する処理を情報処理装置に実行させる実行可能モジュール生成方法であって、

実行可能モジュールに含まれた複数のプロテクトコードの矛盾により処理を終了させる手続きを含むプロテクト対象オブジェクトを、暗号化済みのプロテクト対象オブジェクトの復号により生成し、

互いに関連付けられた第一のプロテクトコードと第二のプロテクトコードとを生成して、前記復号後のプロテクト対象オブジェクトに当該第一のプロテクトコードを埋め込み、当該第一のプロテクトコード埋込み済みのプログラム対象オブジェクトと他のオブジェクトとの結合により実行可能モジュールを作成したら、当該実行可能モジュールに当該第二のプロテクトコードを埋め込み、

前記第一のプロテクトコード埋込み済みのプロテクト対象オブジェクトは、前記二のプロテクトコード埋込み前に消去されることを特徴とする実行可能モジュール生成方法。

【請求項 1 5】

複数のオブジェクトの結合により組み立てられた実行可能モジュールを実行可能な装置が実行するための実行可能モジュールが格納された、機械読み取り可能な記憶媒体であって、

前記複数のオブジェクトのうちの、少なくとも 1 のオブジェクトには、



少なくとも1つのプロテクトコードに矛盾があるか否かを診断し、当該診断結果に応じて、前記実行可能モジュールの処理を終了させる手続きを含んだライブラリオブジェクトが含まれ、

前記実行可能モジュールには、

少なくとも1つのプロテクトコードが埋め込まれている、

ことを特徴とする記憶媒体。

【請求項16】

複数のオブジェクトの結合によって生成された実行可能モジュールを実行するエンタテインメント装置であって、

前記少なくとも1つのオブジェクトに含まれたプロテクトコードと、それ以外のプロテクトコードとが前記実行可能モジュールに含まれている場合、それらのプロテクトコード間の関係を診断し、当該関係に矛盾があれば、前記実行可能モジュールの処理を終了させる手段を有することを特徴とするエンタテインメント装置。

【請求項17】

情報処理装置に実行されるプログラムであって、

前記情報処理装置に、

実行可能モジュールに含まれたプロテクトコードの矛盾により処理を終了させる手続きを含むプロテクト対象オブジェクトを、暗号化済みのプロテクト対象オブジェクトの復号により生成する復号処理と、

前記復号処理で生成されたプロテクト対象オブジェクトと他のオブジェクトとの結合により、前記実行可能モジュールを作成する結合処理と、

前記結合処理による作成される実行可能モジュールに前記プロテクトコードを含ませるためのコード書込み処理と、

前記復号処理で生成されたプロテクト対象オブジェクトを、前記他のオブジェクトとの結合後に消去する削除処理と、

を実行させることを特徴とするプログラム。

【請求項18】

情報処理装置に実行されるプログラムであって、

前記情報処理装置に、

実行可能モジュールに含まれた複数のプロテクトコードの矛盾により処理を終了させる手続きを含むプロテクト対象オブジェクトを、暗号化済みのプロテクト対象オブジェクトの復号により生成する復号処理と、

互いに関連付けられた第一のプロテクトコードと第二のプロテクトコードとを生成するコード生成処理と、

前記復号処理後、当該復号処理で生成されたプロテクト対象オブジェクトに、前記第一のプロテクトコードを埋め込む第一コード書込み処理と、

前記コード書込み処理後、当該コード書込み処理で前記第一のプロテクトコードが埋め込まれたプロテクト対象オブジェクトと、他のオブジェクトとの結合により、実行可能モジュールを作成する結合処理と、

前記結合処理後、当該結合処理で作成された前記実行可能モジュールに前記第二プロテクトコードを埋め込む第二コード書込み処理と、

前記第一コード書込み処理後、前記第二コード書込み処理前に、前記復号処理で生成された前記プロテクト対象オブジェクトを消去する削除処理と、

を実行させることを特徴とするプログラム。

【請求項 1 9】

請求項 1 8 記載のプログラムであって、

前記コード生成処理において、前記情報処理装置に、

乱数から、前記第一のプロテクトコードと前記第二のプロテクトコードを生成させることを特徴とするプログラム。

【請求項 2 0】

請求項 1 8 または 1 9 記載のプログラムであって、

前記情報処理装置に、

前記第一のプロテクトコードと前記第二のプロテクトコードとにダミーデータを付加させることを特徴とするプログラム。

【請求項 2 1】

請求項 1 7、1 8、1 9 および 2 0 のいずれか 1 項に記載のプログラムであって、

前記情報処理装置に、前記実行可能モジュールに含ませるためのプロテクトコードを暗号化する処理を実行させ、

前記プロテクト対象オブジェクトには、前記実行可能モジュールを読み込んだ装置に、当該実行可能モジュールに含まれた暗号化プロテクトコードを、当該プロテクトコードの診断に際して復号させるための手続きが含まれる、

ことを特徴とするプログラム。

【請求項 2 2】

情報処理装置によって処理されるオブジェクトであって、

暗号化され、かつ、

自身を組み込んだ実行可能モジュールを読み込んだ装置に、当該実行可能モジュールに含まれた 1 以上のプロテクトコードに矛盾があれば処理を終了させるための手続きを含む、

ことを特徴とするオブジェクト。

【請求項 2 3】

請求項 2 2 記載のオブジェクトであって、

前記実行可能モジュールに含まれているプロテクトコードが暗号化されている場合に、当該プロテクトコードに矛盾があるか否かの診断に先立ち、当該プロテクトコードを前記装置に復号させるための手続きを含む、ことを特徴とするオブジェクト。

【請求項 2 4】

複数のオブジェクトの結合により組み立てられた実行可能モジュールを実行可能な装置が実行する実行可能モジュールであって、

少なくとも 1 つのプロテクトコードが埋め込まれ、

前記複数のオブジェクトには、

前記装置に、前記実行可能モジュールに含まれているプロテクトコードに矛盾があるか否かを診断させ、当該診断結果に応じて、前記実行可能モジュールの処理を終了させる手続きを含んだライブラリオブジェクトが含まれる、

ことを特徴とする実行可能オブジェクト。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は、プログラム開発環境等におけるリソース不正コピーの防止技術に関する。

【 0 0 0 2 】

【従来の技術】

プログラムの不正コピーを防止するための技術として、特許公報第 2 5 6 9 5 6 4 号記載の、ソフトウェアのコピープロテクト装置が知られている。この装置によれば、フロッピディスクから読み出したプロテクトコードが外部に出力されないため、その不正な書替えを防止することができる。これにより、不正に複製されたソフトウェアを実行可能とすることが困難となり、ソフトウェアの不正な複製を効果的に防止することができる。

【 0 0 0 3 】

【発明が解決しようとする課題】

プログラムが作成される際には、通常、グラフィック機能その他の基本的機能を提供するライブラリ等のリソースが使用される。このような、ライブラリ等のリソースに含まれているオブジェクトは、実行可能モジュールではない。そのため、不正コピー防止のためのプロテクトをライブラリにかけることが困難である。したがって、現在のところ、ライブラリ等のリソースは、メーカー側がプログラム開発者の良識を信頼するというかたちで、プロテクトがかけられていないまま提供されているのが実情である。

【 0 0 0 4 】

しかし、ライブラリ等のリソースは、プログラムと同様、重要な開発成果であるため、なんらかのコピープロテクトがかけられるようにすることが望まれている。

【 0 0 0 5 】

そこで、本発明は、実行可能モジュール以外のオブジェクトの不正コピーを防止することを目的とする。

【 0 0 0 6 】

【課題を解決するための手段】

上記課題を解決するため、本発明は、

実行可能モジュールに含まれたプロテクトコードの矛盾により処理を終了させる手続きを含む、暗号化済みのプロテクト対象オブジェクトを記憶した記憶手段と、

前記記憶手段から前記暗号化済みプロテクト対象オブジェクトを読み出し、当該暗号化済みプロテクト対象オブジェクトを復号する復号手段と、

前記復号後のプロテクト対象オブジェクトと他のオブジェクトとの結合によって作成される実行可能モジュールに前記プロテクトコードを含ませるコード書込み手段と、

前記復号後のプロテクト対象オブジェクトを、前記他のオブジェクトとの結合を行った後に消去する削除手段と、

を有することを特徴とする情報処理装置を提供する。

【 0 0 0 7 】

【発明の実施の形態】

以下、添付の図面を参照しながら、本発明に係る実施の一形態について説明する。ただし、ここでは、ライブラリをプロテクト対象オブジェクトとする。

【 0 0 0 8 】

まず、図 1 により、プログラム開発側で使用される情報処理システムの概略ハードウェア構成について説明する。

【 0 0 0 9 】

プログラム開発側の情報処理システムは、情報処理装置 1 0 0、ユーザ(プログラム開発者)からのデータ入力を受け付ける入力装置(マウス、キーボード等) 1 2 0、各種データを出力する出力装置(ディスプレイ装置等) 1 3 0、等から構成されている。さらに、このシステムには、その他の装置が必要に応じて増設されることもある。

【 0 0 1 0 】

情報処理装置 1 0 0 は、後述の実行可能プログラムファイル作成処理等を実行する CPU 1 0 1、グラフィック描画処理等を実行するグラフィックス機構 1 0

2、主記憶103、CPU101と主記憶103との間をつないだプロセッサバス105、CPU101等と主記憶103またはバス106との間のデータ転送を制御するメモリ／バス制御チップ104、グラフィックス機構102のグラフィックス制御チップとメモリ／バス制御チップ104との間をつないだAGP、ソフトウェア開発のために必要となる各種データ(シリアル番号等の識別情報が埋め込まれたフロントエンドツール、フロントエンドツールによって起動されるリンカおよびコンパイラ、フロントエンドツールとともに提供された暗号化ライブラリファイル等)が格納されたハードディスク106、ハードディスク106に対するデータ読み込み／書き込み処理を実行するハードディスク・ドライブ107、ハードディスクドライブ107がつながれた高速バス106、記憶媒体Aに対するデータ読み込み読書き／書き込み処理を実行するドライブ112、入力装置120からのデータ転送を制御するコントローラ111、出力装置130へのデータ転送を制御するコントローラ113、各コントローラ111,113とドライブ112とがつながれた低速バス109、低速バス109と高速バス106との間をつないだブリッジ回路110、通信ポート(シリアルポート、パラレルポート等)、等を備えている。なお、ここで示した情報処理装置100の内部構成は、ソフトウェア開発に使用されるものとしての一例である。

#### 【0011】

つぎに、この情報処理システムのハードディスク106内の格納データについて説明する。

#### 【0012】

コンパイラは、CPUに、指定ソースプログラムファイルに記述されたソースを、解析処理(字句解析、構文解析、意味解析)等によってオブジェクトとして翻訳編集させ、そのオブジェクトをオブジェクトファイルに格納させる処理が定義されたソフトウェアである。リンカは、CPUに、指定オブジェクトファイル群のそれぞれに記述されているオブジェクトを結合編集させることによって実行可能モジュールを組み立てさせ、その実行可能モジュールを実行可能プログラムファイルに格納させる処理が定義されたソフトウェアである。そして、本実施の形態に係るフロントエンドツールは、CPUに、コンパイラおよびリンカを起動さ

せ、後述の実行可能プログラムファイル作成処理を実行させる処理が定義されたソフトウェアである。

【0013】

暗号化ライブラリファイル $L_1$ （以下、暗号化 $L i b$ ファイル $L_1$ と呼ぶ）は、フロントエンドツールとともに、ユーザ（プログラム開発者）に提供される。例えば、暗号化 $L i b$ ファイル $L_1$ が記録されたインストール用 $C D - R O M$ と、フロントエンドツールが記録されたインストール用フロッピーディスクとがワンセットとしてユーザに提供される。

【0014】

この暗号化 $L i b$ ファイル $L_1$ は、1以上のライブラリファイル $L_2$ （以下、 $L i b$ ファイル $L_2$ と呼ぶ）を適当な暗号化処理で暗号化することによって作成されたものである。ユーザは、実行可能プログラムファイル作成時に、この暗号化 $L i b$ ファイル $L_1$ をフロントエンドツールで復号化することによって、実行可能プログラムファイル作成に必要な1以上の $L i b$ ファイル $L_2$ を得る。このように、ライブラリファイルを、暗号化された状態で提供し、それと共に提供されたフロントエンドツールによってその復号化させるようにしたのは、後述するように、フロントエンドツールを持たないユーザによる $L i b$ ファイル不正使用を禁じるためである。

【0015】

暗号化 $L i b$ ファイル $L_1$ の復号により生成される $L i b$ ファイル $L_2$ には、グラフィック機能、イベント処理等、アプリケーションにおいて使用される基本的な関数等のオブジェクトが部品化されて格納されている。また、すべてのアプリケーションにおいて呼び出される初期化関数が含まれている、 $L i b$ ファイル $L_2$ 内のオブジェクトには、さらに、初期化関数において呼び出されるプロテクトコードチェック手続きと、オブジェクトファイル $M_2$ との結合編集前に第一プロテクトコード $C_1$ が格納される第一プロテクトコード格納変数とが定義されている。ここで、プロテクトコードチェック手続きには、第一プロテクトコード格納変数の格納データとプログラムファイル末尾の格納データとが、正規のプロテクトコード $C_1, C_2$ が満たす関係を満たしているか否かをチェックし、その関係を

満たしていなければ実行中プログラムを初期化段階で終了させる手続きが定義されている。

#### 【0016】

なお、本実施の形態では、初期化関数が含まれているLibファイル $L_2$ だけにプロテクトコードチェック関数および第一プロテクトコード格納変数を含ませているが、すべてのLibファイルにプロテクトコードチェック関数および第一プロテクト格納変数を含ませることにしてもよい。または、代表的なLibファイル、例えば、プロテクトをかける必要のあるLibファイルだけにプロテクトコードチェック関数および第一プロテクト格納変数を含ませることにしてもよい。

#### 【0017】

本実施の形態に係る情報処理システムは、このようなハードウェア構成およびハードディスク格納データ(フロントエンドツール、コンパイラ、リンカ等のソフトウェア、暗号化ライブラリファイル)によって、図2に示すような機能構成を実現する。具体的には、(1)フロントエンドツールの起動コマンドの入力を受け付ける入力受付部200、(2)起動時にフロントエンドツール複製チェック処理を実行する起動時チェック処理部201、(3)開発成果(ソースプログラムファイル $M_1$ 、オブジェクトファイル $M_2$ 、実行可能プログラムファイル $M_3$ )を格納するための開発成果記憶部202、(4)ソースプログラムファイル $M_1$ からオブジェクトファイル $M_2$ を生成するコンパイル処理部203、(5)暗号化Libファイル $L_1$ を記憶したライブラリ記憶部204、(6)暗号化Libファイル $L_1$ を復号する復号化処理部205、(7)復号化処理部204の復号化処理によって生成されたLibファイルを一時ファイル $L_2$ として記憶する一時ファイル記憶部(tmp)206、(8)Libファイル $L_2$ とオブジェクトファイル $M_2$ との結合編集により実行可能プログラムファイル $M_3$ を生成するリンク処理部207、(9)リンク処理部207の結合編集処理後に一時ファイル $L_2$ を削除する一時ファイル削除処理部208、(10)リンク処理部207の結合編集処理前のLibファイル $L_2$ とこのLibファイル $L_2$ から作成された実行可能プログラムファイル $M_3$ とにそれぞれプロテクトコード $C_1, C_2$ を付与するコード付与処理部209、(



1 1) これら各処理部を制御する制御処理部 2 1 0、が実現される。

【 0 0 1 8 】

ここで、コード付与処理部 2 0 9 は、( 1 2 ) 互いに関連付けられた 1 組のプロテクトコード  $C_1, C_2$  (例えば、 $C_1 = f(C_2)$  の関係を満たす 1 組のプロテクトコード  $C_1, C_2$ ) を生成するコード生成処理部 2 0 9 a、( 1 3 ) コード生成処理部 2 0 9 a が生成した 1 組のプロテクトコード  $C_1, C_2$  のうちの一方のコード  $C_1$  (第一プロテクトコードと呼ぶ) を  $L_i b$  ファイル  $L_2$  内のコード格納変数に書き込むとともに他方のコード  $C_2$  (第二プロテクトコードと呼ぶ) を実行可能プログラムファイル  $M_3$  の末尾に付加するコード書込処理部 2 0 9 b、からなっている。

【 0 0 1 9 】

つぎに、これら各機能構成部により実行される実行可能プログラムファイル作成処理について説明する。

【 0 0 2 0 】

プログラム開発者は、コーディングが終了したら、ソースプログラムファイル  $M_1$  をハードディスク 1 0 8 の所定の領域 (図 2 の開発成果記憶部 2 0 2 に相当) に格納してから、そのソースプログラムファイル名と必要なライブラリファイル名とを指定してフロントエンドツールを起動する。これにより、CPU が、フロントエンドツールに定義された処理を実行し、以下に示す実行可能プログラムファイル作成処理 (図 3 のフローチャート参照) が実現される。

【 0 0 2 1 】

まず、フロントエンドツールの起動コマンドを受け付けた入力受付部 2 0 0 は、制御処理部 2 1 0 に処理開始を指示する (S 3 0 0)。制御処理部 2 1 0 は、この指示を受け付けると、フロントエンドツール複製チェック処理の実行を起動時チェック処理部 2 0 1 に指示する。起動時チェック処理部 2 0 1 は、この指示にしたがって、以下に示す方法により、フロントエンドツールの不正使用をチェックする (S 3 0 1)。

【 0 0 2 2 】

例えば、プログラム開発環境に LAN が構築されており、ネットワーク上で起動中のプログラムの識別情報が重複していないか否かを監視するチェックプログ

ラムがネットワーク上に常駐している場合であれば、起動時チェック処理部 2 0 1 は、フロントエンドツールに付与されている識別情報をネットワーク上に送信する。これに応じて、チェックプログラムは、それと同じ識別情報がネットワーク上に存在するか否かを返信する。フロントエンドツールがコピー品であれば、同じ識別情報が複数ネットワーク上に存在し、フロントエンドツールがコピー品でなければ、同じ識別情報が複数ネットワーク上に存在することはないため、フロントエンドツールの不正使用のチェックが可能となる。

## 【 0 0 2 3 】

また、フロントエンドツールに dongle が割り当てられている場合であれば、起動時チェック処理部 2 0 1 は、情報処理装置の通信ポートに装着されている dongle に認証メッセージを送る。このメッセージに応じて、dongle は、自身に割り当てられた識別情報を返信する。認証されたユーザがフロントエンドツールを起動した場合であれば、dongle から返信された識別情報が、フロントエンドツールの識別情報と一致し、認証されたユーザ以外のユーザがフロントエンドツールを起動した場合であれば、dongle の装着がないために識別情報の返信がないか、または、dongle から返信された識別情報がフロントエンドツールの識別情報と一致しないことになるため、フロントエンドツールの不正使用のチェックが可能となる。

## 【 0 0 2 4 】

このようなチェックの結果に基づき、起動時チェック処理部 2 0 1 は、フロントエンドツールの使用に不正 (フロントエンドツールが複製品である、認証ユーザ以外のユーザによる使用である、等) があるか否かを判断する (S 3 0 2)。

## 【 0 0 2 5 】

その結果、起動時チェック処理部 2 0 1 がフロントエンドツールの使用に不正があると判断した場合には、制御処理部 2 1 0 が、エラーメッセージを出力してから (S 3 0 3)、処理を終了させる (S 3 0 8)。これにより、フロントエンドツールの不正使用が禁止されるため、フロントエンドツールの不正使用者には、暗号化 Lib ファイル  $L_1$  を復号することが困難となる。したがって、Lib ファイル  $L_2$  の不正使用も禁止される。

## 【0026】

一方、起動時チェック処理部201がフロントエンドツールの使用に不正がないと判断した場合には、制御処理部210が、指定のソースプログラムファイル $M_1$ の機械語翻訳をコンパイル処理部203に指示する。コンパイル処理部203は、この指示に応じて、指定のソースプログラムファイル $M_1$ をアセンブリプログラムにコンパイルし、そのアセンブリプログラムをオブジェクトにアセンブルする。その後、コンパイル処理部203は、このとき生成したオブジェクトファイル $M_2$ を開発成果記憶部202に格納してから、制御処理部210に正常終了を報告する。

## 【0027】

制御処理部210は、コンパイル処理部203から正常終了報告を受け付けると、暗号化Libファイル $L_1$ の復号化を復号化処理部205に指示する。復号化処理部205は、この指示に応じて、ライブラリ記憶部204から暗号化Libファイル $L_1$ を取り出し、それを復号化する。これにより1以上のLibファイル $L_2$ が生成されると、復号化処理部205は、そのLibファイル $L_2$ を、適当なファイル名の一時ファイルとして一時ファイル記憶部206に格納し、制御処理部210に正常終了を報告する。

## 【0028】

制御処理部210は、復号化処理部205からの正常終了報告を受け付けると、Libファイル $L_2$ に対するプロテクトコード付与をコード付与処理部209に指示する。この指示に応じて、コード付与処理部209では、以下の処理が実行される。まず、コード生成処理部209aが、図4に示すように、乱数Cを発生し、この乱数Cを2つの加工方法 $F_1, F_2$ (例えば、乱数Cを入力データとする2つの関数等)でそれぞれ加工することによって2つのプロテクトコード $C_1, C_2$ を生成する。ここで生成された2つのプロテクトコード $C_1, C_2$ には、前述したように、予め定められた関係(ここでは、 $C_1 = f(C_2)$ とする)がある。これらのプロテクトコード $C_1, C_2$ は、さらに、コード生成処理部209aによって、適当な暗号化方法でそれぞれ暗号化される。ついで、コード書込み処理部209bが、第二プロテクトコード $C_2$ を保持するとともに、所定のLibファイル $L_2$ 内

の第一プロテクトコード格納変数に第一プロテクトコード $C_1$ を格納する(S 3 0 4)。

【0 0 2 9】

このようにして、L i bファイル $L_2$ に対するプロテクトコード付与が終了すると、制御処理部 2 1 0 は、リンク処理部 2 0 7 に、オブジェクトファイル $M_2$ と指定のL i bファイル $L_2$ との結合編集を指示する。リンク処理部 2 0 7 は、この指示に応じて、オブジェクトファイル $M_2$ と指定のL i bファイル $L_2$ とを結合し、実際の番地の埋込み等を行う。これにより実行可能プログラムファイル $M_3$ が組み立てられると、リンク処理部 2 0 7 は、その実行可能プログラムファイル $M_3$ を、適当な名前で開発成果記憶部 2 0 2 に格納してから、正常終了を制御処理部 2 1 0 に報告する(S 3 0 5)。

【0 0 3 0】

制御処理部 2 1 0 は、リンク処理部 2 0 7 からの正常終了報告を受け付けると、一時ファイル削除を一時ファイル削除処理部 2 0 8 に指示する。この指示に応じて、一時ファイル削除処理部 2 0 8 は、すべてのL i bファイル $L_2$ を一時ファイル記憶部 2 0 6 から削除する(S 3 0 6)。

【0 0 3 1】

その後、制御処理部 2 1 0 は、実行可能プログラムファイル $L_3$ に対するプロテクトコード付与をコード付与処理部 2 0 9 に指示する。この指示に応じて、コード付与処理部 2 0 9 のコード書込み処理部 2 0 9 b が、保持中の第二プロテクトコード $C_2$ を実行可能プログラムファイル $M_3$ に付加してから、正常終了を制御処理部 2 1 0 に報告する(S 3 0 7)。なお、ライセンシー管理(例えば、プログラム作成会社の識別等)のために、このときフロントエンドツールの識別情報も、、コード付与処理部 2 0 9 によって実行可能プログラムファイル $M_3$ に埋め込まれるようにしてもよい。

【0 0 3 2】

制御処理部 2 1 0 は、コード付与処理部 2 0 9 からの正常終了報告を受け付けると、処理を終了させる(S 3 0 8)。なお、以上の処理により作成された実行可能プログラムファイル $M_3$ は、例えば、光ディスク等の記憶媒体に焼き付けられ

て市場に供給される(図9参照)。

### 【0033】

本実施の形態に係る実行可能プログラムファイル作成処理によれば、以下に示すように、Libファイル $L_2$ の不正使用が防止される。

(1)ハードディスクに格納されたLibファイルが暗号化されており、かつ、それを復号するフロントエンドツールの不正使用が、起動時チェック(図2のS301)によって禁止される。このため、フロントエンドツールの正当使用者以外のユーザによる、Libファイルの使用が防止される。

(2)図5に示すように、実行中のプログラムファイルに埋め込まれたプロテクトコードに矛盾があった場合にその処理を終了させるプロテクトコードチェック手続きEをLibファイル $L_2$ にあらかじめ含めておき、フロントエンドツールの使用によってLibファイル $L_2$ とオブジェクトファイル $M_2$ とが結合された実行可能プログラムファイル $M_3$ にだけ、実行可能プログラムファイル作成処理の過程(S304、S307)において適正なプロテクトコード $C_1, C_2$ が埋め込まれるようにすることによって、フロントエンドツールの正当使用により作成された実行可能プログラムファイル $M_3$ のみを実行可能とすることができる。すなわち、フロントエンドツールによらずLibファイルが結合された実行可能プログラムファイル、具体的には、不正なプロテクトコード(一方または両方)が埋め込まれた実行可能プログラムファイル、プロテクトコード(一方または両方)が埋め込まれていない実行可能プログラムファイル等の実行が、初期化段階において阻止される。このため、フロントエンドツールの正当使用者以外のユーザによる、Libファイルの使用が防止される。

(3)図5に示すように、乱数Cから生成された1組のプロテクトコード $C_1, C_2$ のうち、第一プロテクトコード $C_1$ だけをLibファイル $L_2$ に埋め込んでおき(S304)、その後、Libファイル $L_2$ とオブジェクトファイル $M_2$ との結合編集(S305)が終了したら、実行可能プログラムファイル $M_2$ へ第二プロテクトコード $C_2$ を埋め込む前にLibファイル $L_2$ を消去(S306)してしまうため、使用済みLibファイル $L_2$ の不正利用が防止される。すなわち、一方のプロテクトコード $C_1$ が埋め込まれた使用済みLibファイル $L_2$ を消去前に取り出すこ

とができたとしても、その段階(S 3 0 4 ~ S 3 0 6)では、乱数Cから生成された他方のプロテクトコードC<sub>2</sub>の取り出しが不可能であるから、その使用済みL i bファイルL<sub>2</sub>に埋め込まれている第一プロテクトコードC<sub>1</sub>と矛盾しない第二プロテクトコードC<sub>2</sub>を知ることは困難である。このため、消去前に取り出した使用済みL i bファイルL<sub>2</sub>とオブジェクトファイルとを結合編集したとしても、使用済みL i bファイルL<sub>2</sub>に埋込済みの第一プロテクトコードと適正な関係を有する第二プロテクトコードを実行可能プログラムファイルに埋め込むことは困難である。したがって、使用済みL i bファイルの不正使用が防止される。

(4)最終的に作成された実行可能プログラムファイルM<sub>3</sub>の解析によって、それに埋め込まれた1組のプロテクトコードC<sub>1</sub>, C<sub>2</sub>を知ってから、つぎのフロントエンドツール起動時に使用済みL i bファイルL<sub>2</sub>を消去前に取り出したとしても、プロテクトコードC<sub>1</sub>, C<sub>2</sub>は、毎回、乱数から生成されるため、消去前に取り出した使用済みL i bファイルL<sub>2</sub>に埋め込まれている第一プロテクトコードC<sub>1</sub>に対応する第二プロテクトコードは、実行可能プログラムファイルM<sub>3</sub>の解析によって知った第二プロテクトコードC<sub>2</sub>とは必ずしも一致しない。このため、最終的に作成された実行可能プログラムファイルM<sub>3</sub>の解析によっても、適正な関係を有する1組のプロテクトコードを得ることは困難である。したがって、消去前に取り出した使用済みL i bファイルL<sub>2</sub>とオブジェクトファイルとを結合編集したとしても、使用済みL i bファイルL<sub>2</sub>に埋込済みの第一プロテクトコードと適正な関係を有する第二プロテクトコードを実行可能プログラムファイルに埋め込むことは困難である。このため、使用済みL i bファイルの不正使用が防止される。

#### 【 0 0 3 4 】

以上の4つの防止策によってL i bファイル不正使用が防止されるが、ライブラリバージョンアップのたびに、L i bファイルの暗号化方法とプロテクトコードチェック手続きの内容とをそれぞれ変更するようにすれば、L i bファイルの不正使用防止が強化される。なお、このようにする場合には、L i bファイル複合方法および乱数加工方法が変更されたフロントエンドツールが、新バージョンの暗号化L i bファイルとともにリリースされる必要がある。

## 【 0 0 3 5 】

ところで、以上においては、L i b ファイル  $L_2$  内で定義された第一プロテクトコード格納変数に第一プロテクトコード  $C_1$  を格納し、実行可能プログラムファイル  $M_3$  の末尾に第二プロテクトコード  $C_2$  を付加することとしているが、必ずしも、このようにする必要はない。例えば、プロテクトコード付与処理部が、図 6 に示すように、L i b ファイル  $L_2$  の末尾に、ダミーデータと第一プロテクトコード  $C_1$  とを含んだ第一ダミーエリア①を付加し、実行可能プログラムファイル  $M_3$  の末尾に、ダミーデータと第二プロテクトコード  $C_2$  とを含んだ第二ダミーエリア②を付加するようにしてもよい。そして、2つのダミーエリア①②の合計データ量を変えないように、各ダミーエリア①②に含まれるダミーデータのデータ量をランダムに変化させるようにすれば、1つの実行可能プログラムファイル  $M_3$  を作成するごとに、その内部におけるプロテクトコード  $C_1, C_2$  のアドレスが変化するため、プロテクトコード  $C_1, C_2$  の間にある関係の解析を困難にすることができる。

## 【 0 0 3 6 】

また、以上においては、プロテクトコードチェック手続きが組み込まれた L i b ファイルに第一プロテクトコードをリンク前に埋め込み、実行可能プログラムファイルが完成したときに、さらに、この実行可能プログラムファイルに第二プロテクトコードを埋め込むこととしているが、かならずしも、このようにする必要はない。例えば、プロテクトコードチェック手続きが組み込まれた L i b ファイルにはプロテクトコードを組み込まず、実行可能プログラムファイルが完成したときに、この実行可能プログラムファイルに、プロテクトコードチェック手続きによって正当性をチェック可能なプロテクトコード(例えば、互いに関連付けられた複数のプロテクトコード)を埋め込むようにしてもよい。

## 【 0 0 3 7 】

つぎに、本実施の形態に係るプログラムファイル作成処理により作成された実行可能プログラムファイル  $M_3$  を実行することができるシステムの概略ハードウェア構成について説明する。ただし、ここでは、システムの一例として、エンタテインメント装置を挙げることにする。

## 【 0 0 3 8 】

図 7 は、本実施の形態に係るエンタテインメント装置 7 0 0 およびその周辺機器の外観図であり、図 8 は、本実施の形態に係るエンタテインメント装置 7 0 0 の概略ハードウェア構成図である。

## 【 0 0 3 9 】

本実施の形態に係るエンタテインメント装置 7 0 0 は、CD-ROM、DVD-ROM等の光ディスク 9 0 0 からゲームプログラム(前述のプログラムファイル作成処理により作成された実行可能プログラムファイルM<sub>3</sub>等)を読み出し、そのゲームプログラムにしたがってゲームを実行するものである。ここで、「ゲームの実行」とは、エンタテインメント装置 7 0 0 が、ユーザ(以下、ゲームプレイヤーと呼ぶ)からの指示にしたがって、表示装置(CRT、LCD、プロジェクション装置等)上の表示映像およびオーディオ装置からの音声をゲームストーリー等にあわせて変化させることをいう。

## 【 0 0 4 0 】

図 7 に示すように、エンタテインメント装置 7 0 0 の筐体前面部には、ゲームプレイヤーがゲームを実行させるために必要となる各種の操作部が設けられている。このような操作部の具体例としては、ゲームプログラムが格納された光ディスク 9 0 0 が装着されるディスク装着部 7 0 1、ディスク装着部 7 0 1 からトレイを引き出すためのトレイ操作ボタン 7 0 1 A、ゲームプレイヤーからの入力を受け付ける各種入力受け部(ボタン 7 5 3 A, 7 5 3 B, 7 5 4 ~ 7 5 8、方向キー 7 5 2、スティック 7 5 1 A, 7 5 1 B 等)を有するコントローラ 7 5 0 が接続される複数(ここでは 2 つ)のコントローラ接続部 7 0 2 A, 7 0 2 B、実行中のゲームをリセットするためのリセットボタン 7 0 3、ゲームデータ等を記録させるためのメモ리카ード 7 3 0 が装着される複数(ここでは 2 つ)のメモ리카ード装着部 7 0 4 A, 7 0 4 B、等が挙げられる。ここでコントローラ接続部 7 0 2 A, 7 0 2 B が複数設けられているのは、各コントローラ接続部 7 0 2 A, 7 0 2 B にそれぞれコントローラを接続することによって、エンタテインメント装置 7 0 0 が、複数のゲームプレイヤーからの指示を受け付けることができるようにするためである。



## 【 0 0 4 1 】

また、エンタテインメント装置 7 0 0 の筐体背面部(前面の反対側)には、電源オンオフスイッチ(不図示)、表示装置等が接続される A V 端子(不図示)が設けられている。

## 【 0 0 4 2 】

そして、このエンタテインメント装置 7 0 0 の内部には、図 8 に示すように、装置全体を制御するメイン CPU 8 0 0、A V 端子(不図示)から出力されるビデオ信号を生成するグラフィックプロセッサ(G P) 8 0 1、コントローラ接続部 7 0 2 A、7 0 2 B に接続されたコントローラ 7 5 0 およびメモ리카ード接続部 7 0 4 A、7 0 4 B に接続されたメモ리카ード 7 3 0 とのデータ転送を制御する I/O プロセッサ(I O P) 8 0 2、ディスク装着部 7 0 1 に装着された光ディスク 9 0 0 からのデータ読出しを制御する光ディスク制御部 8 0 3、A V 端子(不図示)から出力されるオーディオ信号を生成するサウンド再生処理プロセッサ(S P U) 8 0 4、サウンドバッファ 8 0 5、メイン CPU 8 0 0 および I O P 8 0 2 が実行する各オペレーティングシステムプログラムが格納された OS-ROM 8 0 6、I O P のワークエリアとして用いられる I O P メモリ 8 0 7、メイン CPU 8 0 0 のワークエリアおよびバッファとして用いられるメインメモリ 8 0 8、これら各部の間をつないだバス 8 0 9 等が搭載されている。

## 【 0 0 4 3 】

このエンタテインメント装置 7 0 0 に電源が投入されると、2 種類のオペレーティングシステムプログラム(メイン CPU 用、I O P 用)がそれぞれ OS-ROM 8 0 6 から読みだされ、それらがメイン CPU 8 0 0 と I O P 8 0 2 とによってそれぞれ起動される。これにより、オペレーティングシステムによる、装置各部の統合制御が開始され、エンタテインメント装置としての各種機能がゲームプレイヤに提供される。具体的には、光ディスクからの実行可能プログラムファイル読み込みおよびその実行、コントローラを介したゲームプレイヤからの指示受け、ゲームプレイヤの指示に応じた映像表示および効果音・楽音出力等が可能な環境が提供される。

## 【 0 0 4 4 】

このような環境において、図 9 に示した光ディスク 900 から、上述の実行可能プログラムファイル作成処理によって作成されたゲーム用実行可能プログラムファイル  $M_3$  が読み込まれると、そのゲーム用実行可能プログラムファイル  $M_3$  に定義された処理を実行するための機能構成が、メイン CPU 800 によってプロセスとして実現される。すなわち、図 10 に示すように、エンタテインメント装置 700 は、(1) 実行可能プログラムファイル  $M_3$  の所定領域(第一プログラム格納変数)の格納データ(実行可能プログラムファイル  $M_3$  が Lib ファイルの正当使用により生成されたものであれば、正当な第一プロテクトコード  $C_1$ ) が格納された第一プロテクトコード格納部 1000、(2) 実行可能プログラムファイル  $M_3$  の末尾領域の格納データ(実行可能プログラムファイル  $M_3$  が Lib ファイルの正当使用により生成されたものであれば、正当な第二プロテクトコード  $C_2$ ) が格納された第二プロテクトコード格納部 1001、(3) 起動時にプロテクトコードチェック手続きを実行するプロテクトコードチェック処理部 1002、(4) プロテクトコードチェック処理部 1002 からの開始指示に応じて、ゲームを実行するゲーム実行処理部 1003、(5) コントローラからの入力信号をゲーム実行処理部 1003 に渡す入力データ受付処理部 1004、(6) ゲーム実行処理部 1003 の指示にしたがって、効果音等のオーディオ信号を生成する音声制御処理部 1005、(7) ゲーム処理部 1003 の指示にしたがって、ゲーム画面映像のビデオ信号を生成する映像制御処理部 1006、が実現される。ここで、プロテクトコードチェック処理部 1002 には、(8) 第一プロテクトコード  $C_1$  および第二プロテクトコード  $C_2$  を復号するコード復号処理部 1002a、(9) 復号後の 2 つのプロテクトコード  $C_1, C_2$  の整合性をチェックするコードマッチング処理部 1002b、(10) 与えられた終了指示にしたがって、処理を終了させる終了処理部 1002d、(11) コードマッチング処理部 1002b のチェック結果に応じて、終了処理部 1002d に終了指示またはゲーム処理部に開始指示を与える制御処理部 1002c、が含まれている。

#### 【0045】

つぎに、図 11 により、これらの機能構成部によって実行される処理について説明する。

## 【 0 0 4 6 】

図 9 の光ディスク 9 0 0 から読み込まれた実行可能プログラムファイル  $M_3$  が起動されると (S 1 1 0 0)、図 1 0 の機能構成部が以下の処理を開始する。

## 【 0 0 4 7 】

ゲームの開始前に、実行可能プログラムファイル  $M_3$  に組み込まれた  $L i b$  ファイル  $L_2$  が正当使用によるものか否かを判断するため、以下の自己診断処理がプロテクトコードチェック処理部 1 0 0 2 によって実行される。

## 【 0 0 4 8 】

まず、制御処理部 1 0 0 2 c が、コードの復号をコード復号処理部 1 0 0 2 a に指示する。この指示に応じて、コード復号処理部 1 0 0 2 a は、第一プロテクトコード格納部 1 0 0 0 の格納データと、第二プログラムコード格納部 1 0 0 1 の格納データとを取り出し、それらをそれぞれ復号する。

## 【 0 0 4 9 】

フロントエンドツールの正当使用により作成された実行可能プログラムファイル  $M_3$  であれば、1 組のプロテクトコード生成に用いられた関係 (ここでは  $C_1 = F(C_2)$ ) を満たすデータ  $C_1, C_2$  が、このときの復号処理によって得られるはずである。そこで、コードマッチング処理部 1 0 0 2 b は、ここで得られた 2 つのデータがその関係を満たすか否かを診断し、その診断結果を制御処理部 1 1 0 2 c に通知する (S 1 1 0 1)。

## 【 0 0 5 0 】

制御処理部 1 1 0 2 c は、実行可能プログラムファイル  $M_3$  に組み込まれた  $L i b$  ファイル  $L_2$  が正当使用によるものか否かを、コードマッチング処理部 1 0 0 2 b の診断結果に基づき判断する (S 1 1 0 2)。具体的には、1 組のプロテクトコード生成に用いた関係を満たさないという診断結果が通知された場合には、プロテクトコードに矛盾ありと判断し、1 組のプロテクトコード生成に用いた関係を満たすという診断結果が通知された場合には、プロテクトコードに矛盾なしと判断する。

## 【 0 0 5 1 】

以上の自己診断処理によってプロテクトコードに矛盾ありと判断した場合には

、制御処理部 1 1 0 2 c は、その旨を表すエラーメッセージを出力し (S 1 0 0 3)、終了処理部 1 0 0 2 d に終了指示を与える。この指示に応じて、終了処理部 1 0 0 2 d は、所定の終了処理を実行し、プログラムを終了させる (S 1 1 0 6)。

#### 【 0 0 5 2 】

一方、プロテクトコードに矛盾なしと判断した場合には、制御処理部 1 0 0 2 d は、ゲーム実行処理部 1 0 0 3 に開始指示を与える。この指示に応じて、ゲーム実行処理部 1 0 0 3 は、ゲーム処理を開始する (S 1 1 0 4)。

#### 【 0 0 5 3 】

以後、コントローラを介して送られてくる、ゲームプレイヤーからの指示は、入力データ受付処理部 1 0 0 4 からゲーム実行処理部 1 0 0 3 に渡される。そして、ゲーム実行処理部 1 0 0 3 は、入力データ受付処理部 1 0 0 4 から渡された指示に応じて、表示装置上の表示映像およびオーディオ装置からの音声を変化させるべく、音声制御処理部 1 0 0 5 と映像制御処理部 1 0 0 6 とを制御する。この間、ゲーム実行処理部 1 0 0 3 は、ゲームオーバーの条件を満たしたか否かの判断 (S 1 1 0 5) を定期的に行ない、ゲームオーバーと判断したら、ゲームを終了させる (S 1 1 0 6)。

#### 【 0 0 5 4 】

このように、本実施の形態に係るエンタテインメント装置によれば、読み込んだゲーム用実行可能プログラムファイルに正当なプロテクトコードが埋め込まれている場合にだけゲームを開始させ、読み込んだゲーム用実行可能プログラムファイルに正当なプロテクトコードが埋め込まれていなければ、ゲームを終了させる。したがって、フロントエンドツールの正当使用によってゲーム用実行可能プログラムファイルが作成されていなければ、エンタテインメント装置でゲームをすることができない。このため、フロントエンドツールの正当使用者以外のユーザによる L i b ファイル使用を止めることができる。

#### 【 0 0 5 5 】

なお、以上においては、上述の実行可能プログラムファイル作成処理により作成された実行可能プログラムファイルを実行するための装置としてエンタテイン

メント装置を例に挙げたが、上述の実行可能プログラムファイル作成処理により作成された実行可能プログラムファイルを実行するための装置は、必ずしもエンタテインメント装置である必要はない。例えば、プログラムの実行能力のある、一般的な電子計算機であってもよい。もちろん、実行可能プログラムファイルも、ゲーム以外の処理を実行させるものであってもよい。

【 0 0 5 6 】

また、ここでは、記憶媒体から実行可能プログラムファイルが実行装置に読み込まれるようにしているが、ネットワーク等の伝送媒体によって実行可能プログラムファイルが伝送され、それが実行装置のハードディスクに格納されるようにしてもよい。

【 0 0 5 7 】

また、以上においては、ライブラリファイルをプロテクト対象としているが、その他のファイル(例えば、画像データ等が格納されたファイル等の、実行可能ファイル以外のファイル)をプロテクト対象としてもよい。

【 0 0 5 8 】

【発明の効果】

本発明によれば、ライブラリ等、実行可能ファイル以外のファイルの不正コピーを防止することができる。

【図面の簡単な説明】

【図 1】

本発明の実施の一形態に係る情報処理システムのハードウェア構成図である。

【図 2】

本発明の実施の一形態に係る情報処理システムの機能構成図である。

【図 3】

本発明の実施の一形態に係る実行可能プログラム作成処理のフローチャートである。

【図 4】

本発明の実施の一形態に係る 1 組のプロテクトコードの生成方法を概念的に示した図である。

【図 5】

本発明の実施の一形態に係る実施可能プログラムファイルが作成されるまでの処理を概念的に示した図である。

【図 6】

本発明の実施の一形態に係る実施可能プログラムファイルのデータ構造を概念的に示した図である。

【図 7】

本発明の実施の一形態に係るエンタテインメント装置およびその周辺機器の外観図である。

【図 8】

本発明の実施の一形態に係るエンタテインメント装置のハードウェア構成図である。

【図 9】

本発明の実施の一形態に係る光ディスクの格納データのデータ構造を概念的に示した図である。

【図 1 0】

本発明の実施の一形態に係るエンタテインメント装置の機能構成図である。

【図 1 1】

本発明の実施の一形態に係る実行可能プログラム作成処理のフローチャートである。

【符号の説明】

$L_1$ …暗号化ライブラリファイル	$L_2$ …ライブラリファイル
$M_1$ …ソースファイル	$M_2$ …オブジェクトファイル
$M_3$ …実行可能ファイル	
1 0 0 …情報処理装置	1 0 1 …C P U
1 0 2 …グラフィックス機構	1 0 3 …主記憶
1 0 4 …メモリバス制御チップ	1 0 5 , 1 0 6 , 1 0 9 …バス
1 0 7 …ハードディスクドライブ	1 0 8 …ハードディスク
1 1 0 …ブリッジ回路	1 1 1 …入力装置コントローラ

1 1 2 … C D - R O M ド ラ イ ブ

1 2 0 … 入 力 装 置

1 1 3 … 出 力 装 置 コ ン ト ロ ー ラ

1 3 0 … 出 力 装 置

2 0 0 … 入 力 受 付 部

2 0 2 … 開 発 成 果 記 憶 部

2 0 4 … ラ イ ブ ラ リ 記 憶 部

2 0 6 … 一 時 ファ イ ル 記 憶 部

2 0 8 … 一 時 ファ イ ル 削 除 処 理 部

2 1 0 … 制 御 処 理 部

2 0 1 … 起 動 時 チ ェ ッ ク 処 理 部

2 0 3 … コ ン パ イ ル 処 理 部

2 0 5 … 復 号 化 処 理 部

2 0 7 … リ ン ク 処 理 部

2 0 9 … コ ー ド 付 与 処 理 部

7 0 0 … エ ン タ テ イ ン メ ン ト 装 置

8 0 1 … グ ラ フ ィ ッ ク プ ロ セ ッ サ

8 0 3 … 光 デ ィ ス ク 制 御 部

8 0 5 … サ ウ ン ド バ ッ フ ァ

8 0 7 … I O P メ モ リ

8 0 9 … バ ス

8 0 0 … メ イ ン C P U

8 0 2 … I / O プ ロ セ ッ サ

8 0 4 … サ ウ ン ド 再 生 処 理 プ ロ セ ッ サ

8 0 6 … O S - R O M

8 0 8 … メ イ ン メ モ リ

1 0 0 0 … 第 一 プ ロ テ ク ト コ ー ド 格 納 部

1 0 0 1 … 第 二 プ ロ テ ク ト コ ー ド 格 納 部

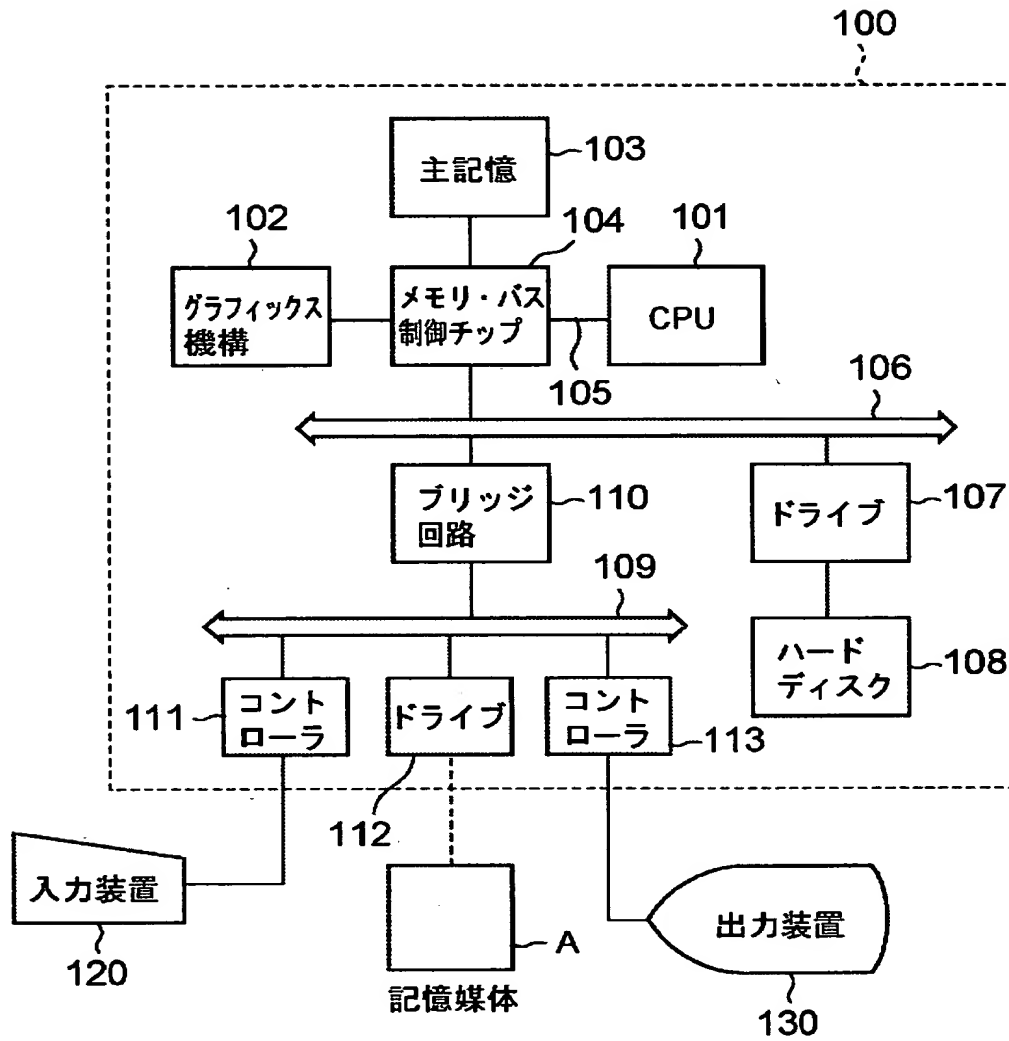
1 0 0 2 … プ ロ テ ク ト コ ー ド チ ェ ッ ク 処 理 部

1 0 0 3 … ゲ ー ム 実 行 処 理 部

【書類名】 図面

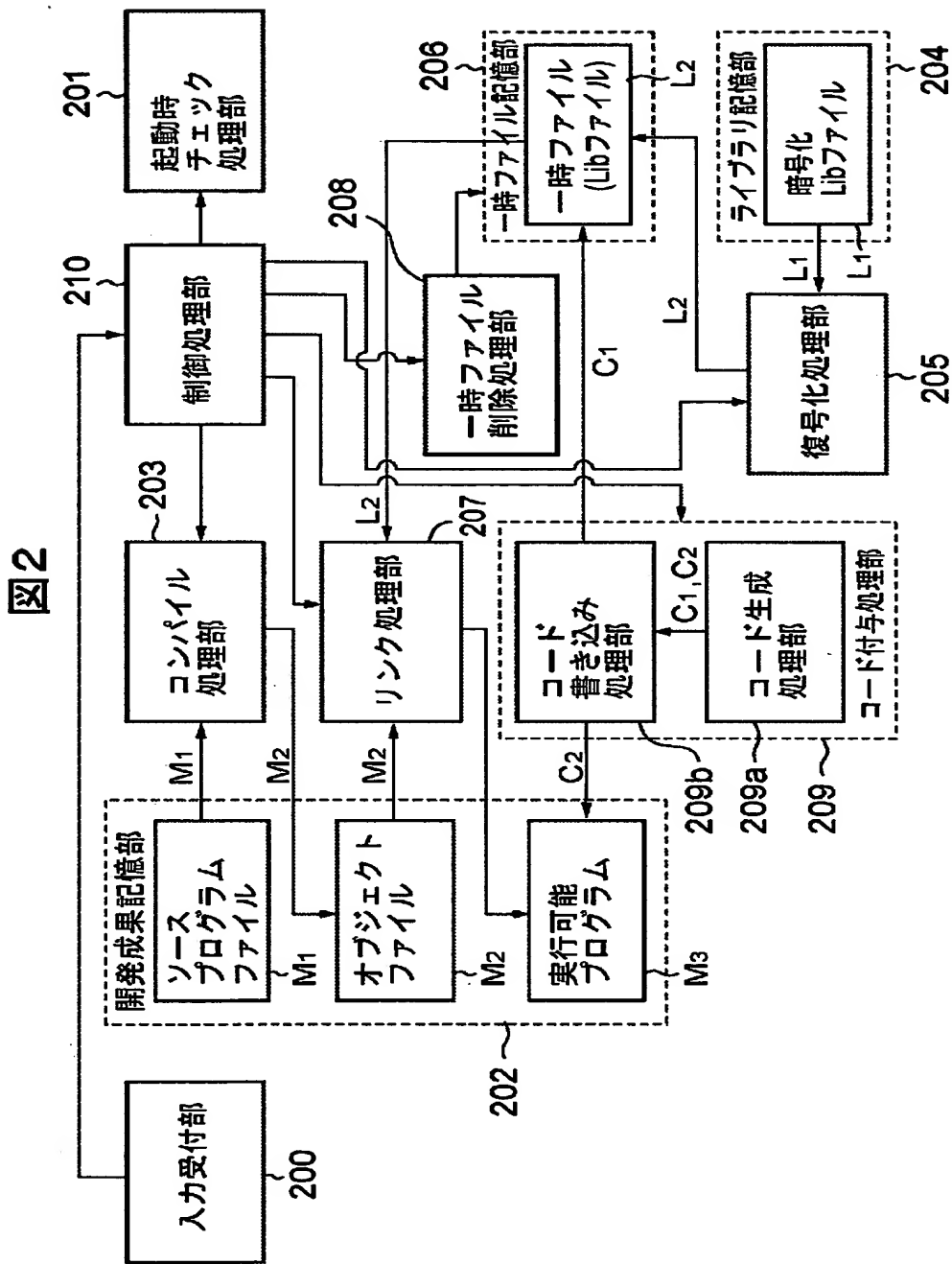
【図 1】

図 1

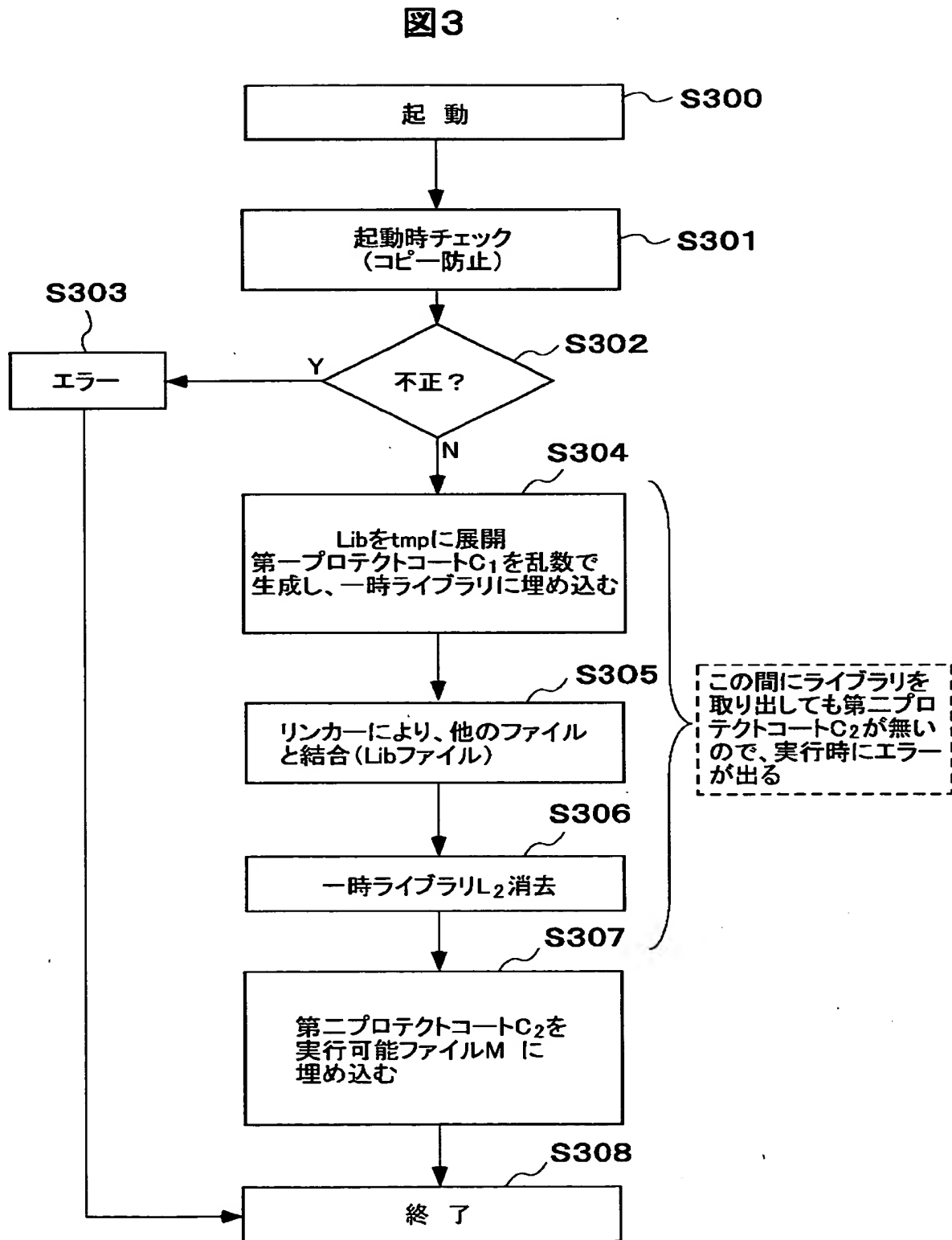




【図2】

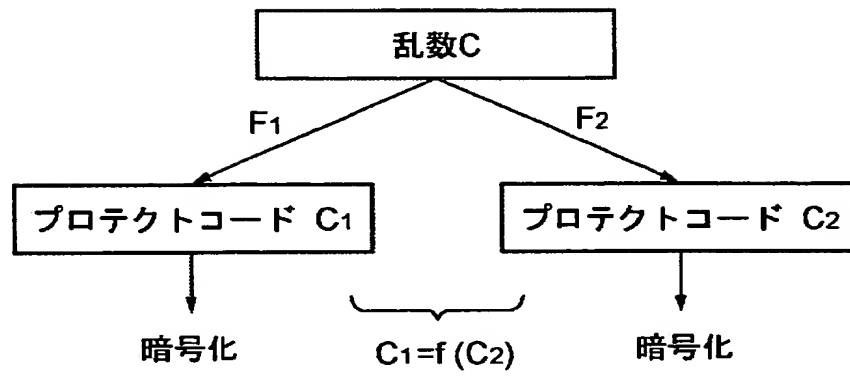


【図 3】

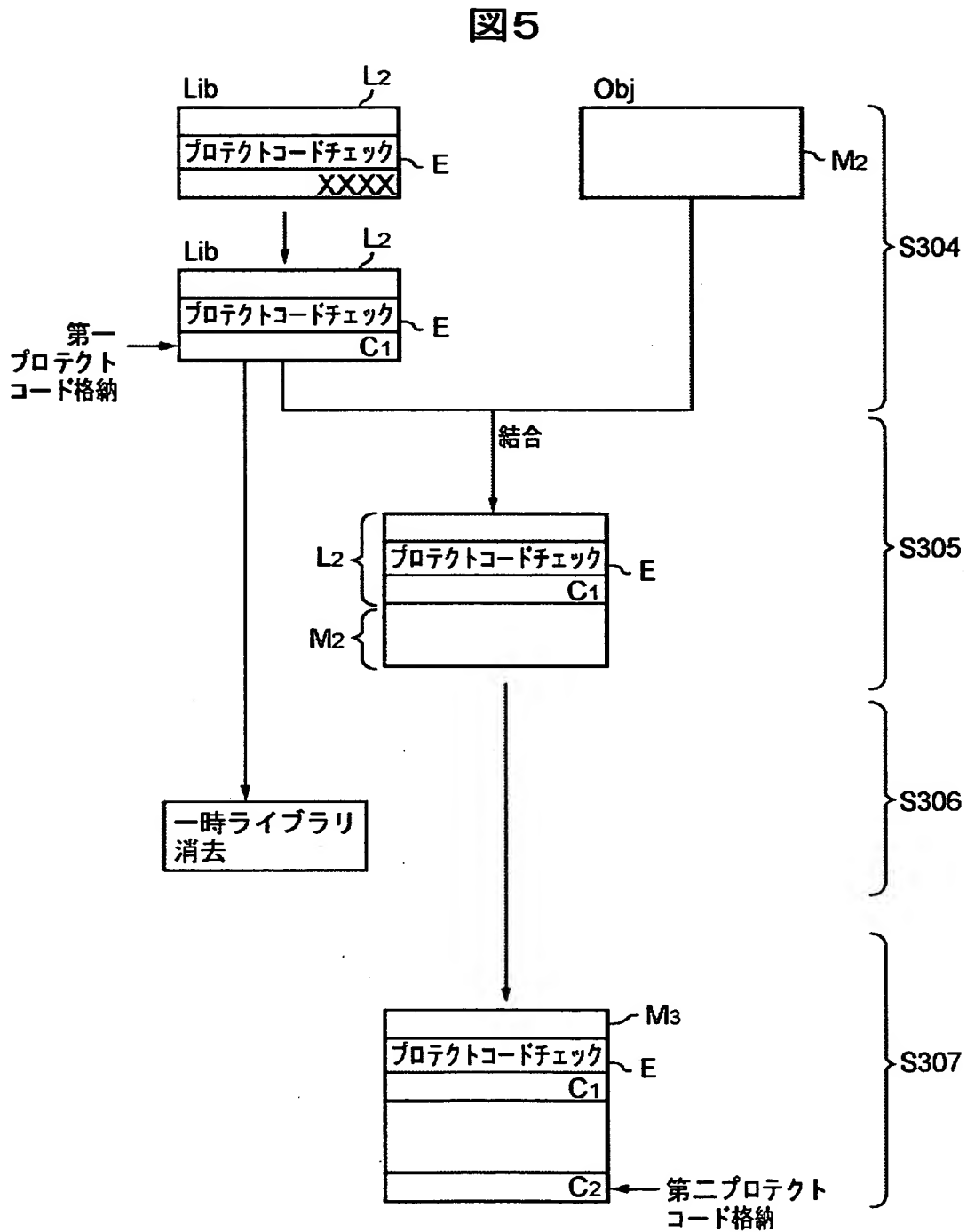


【図 4】

図 4

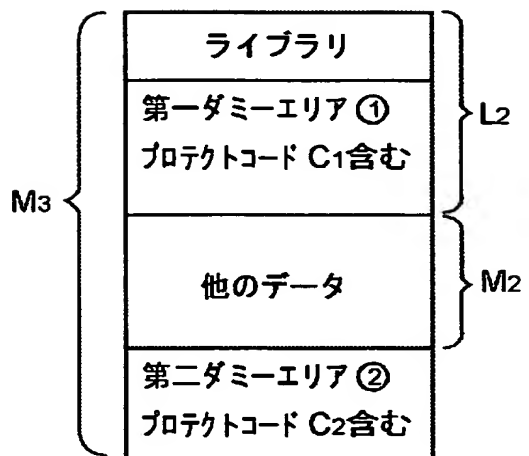


【図5】



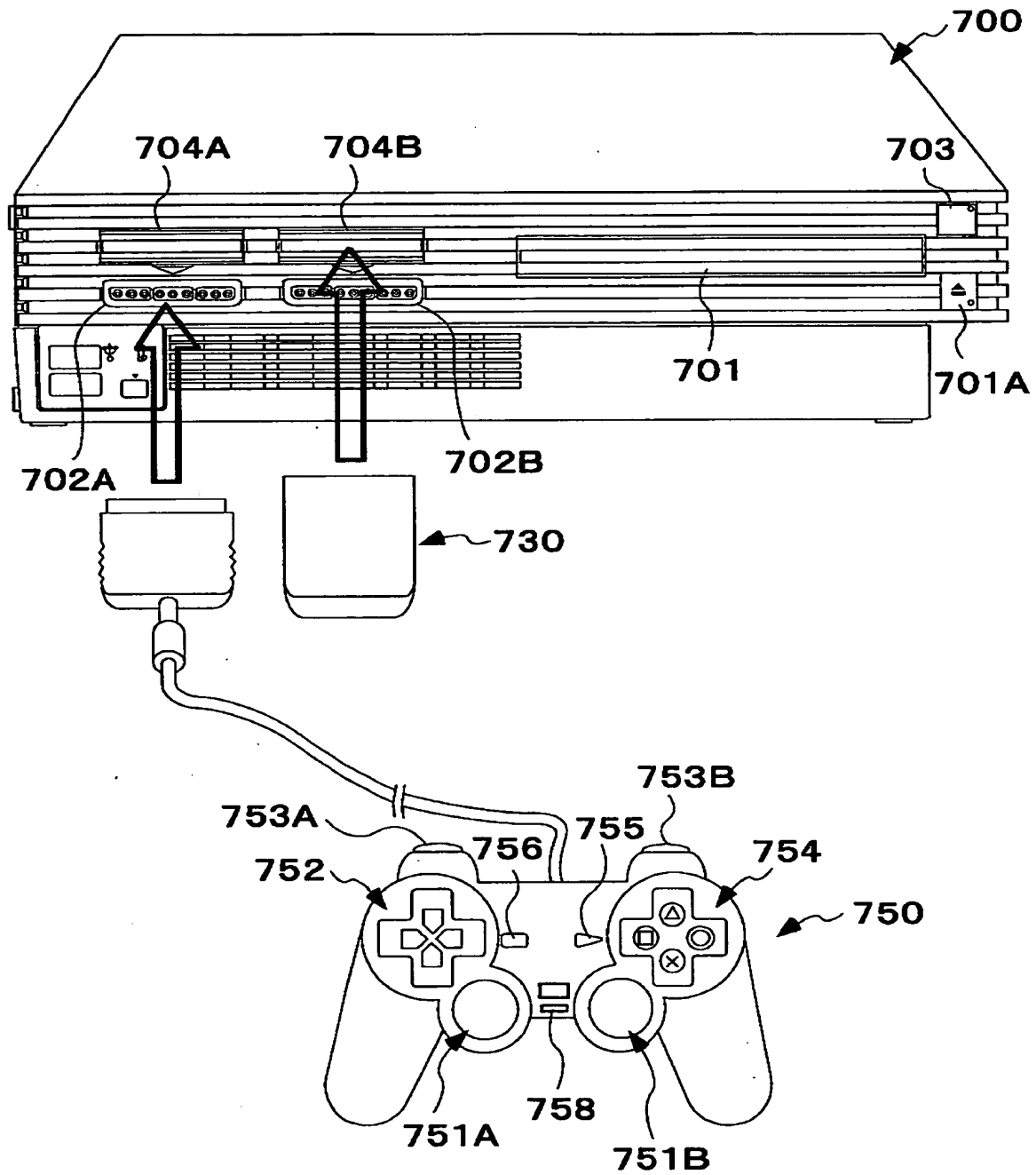
【図 6】

図 6

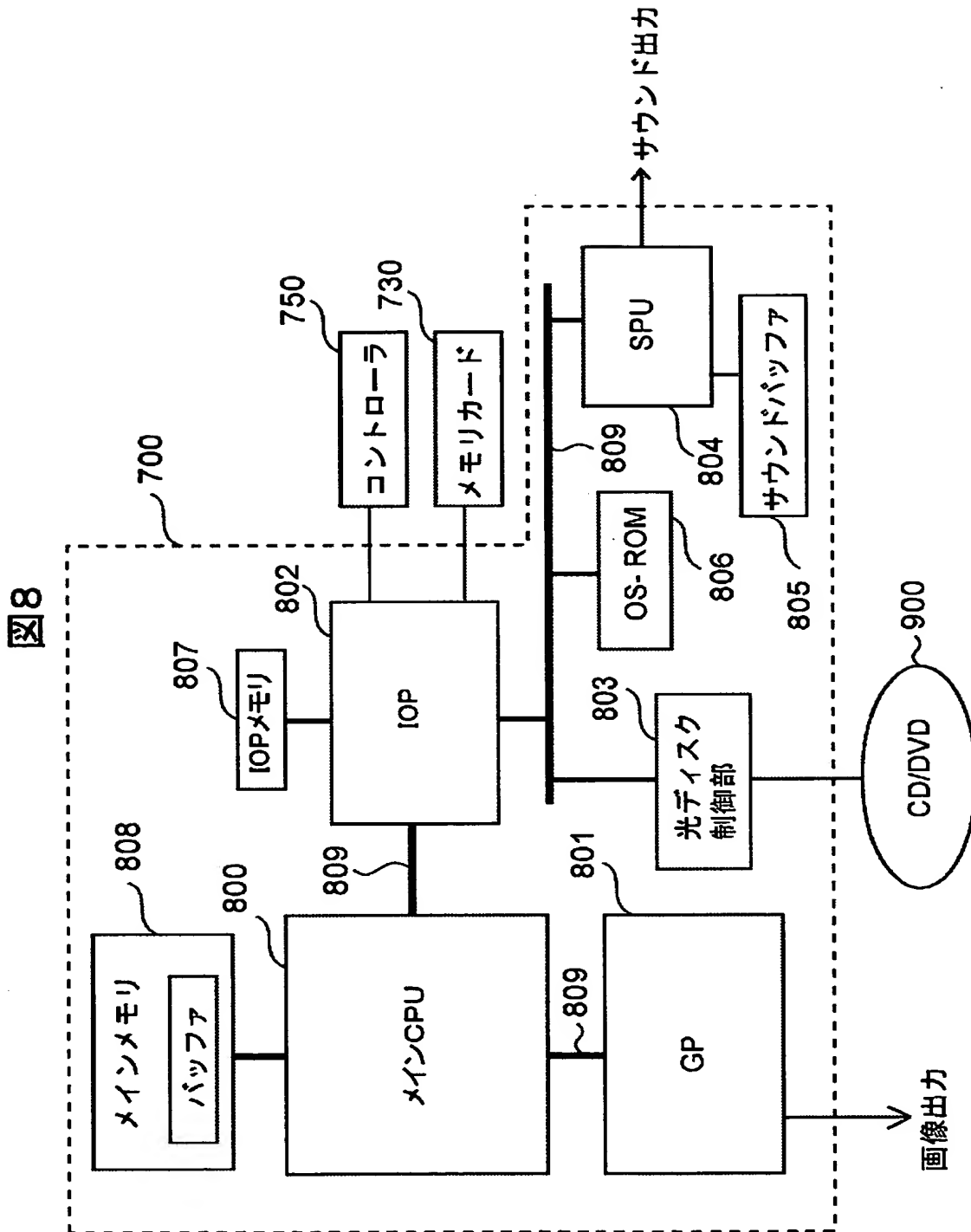


【図 7】

図 7

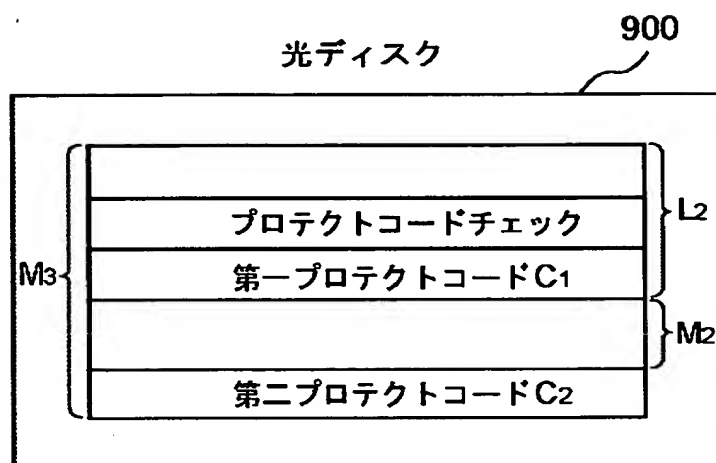


【図 8】



【図 9】

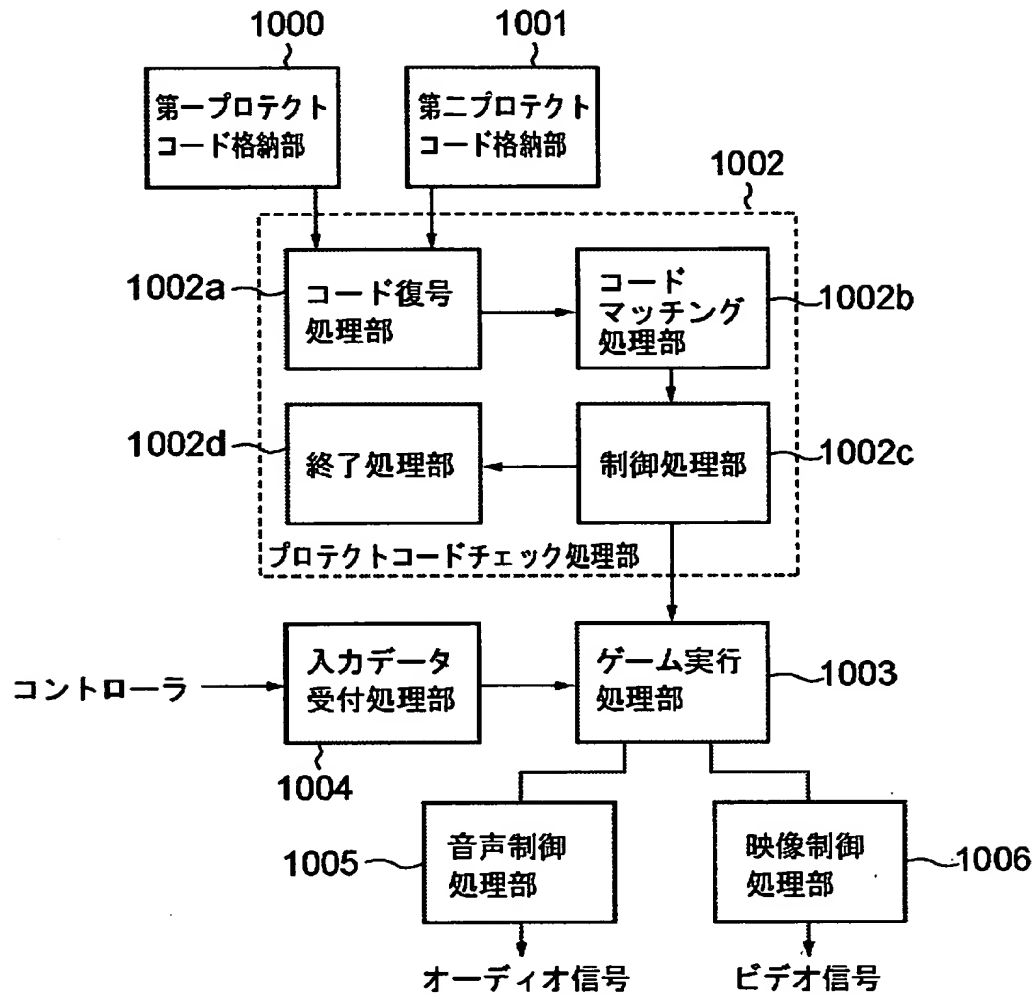
図 9





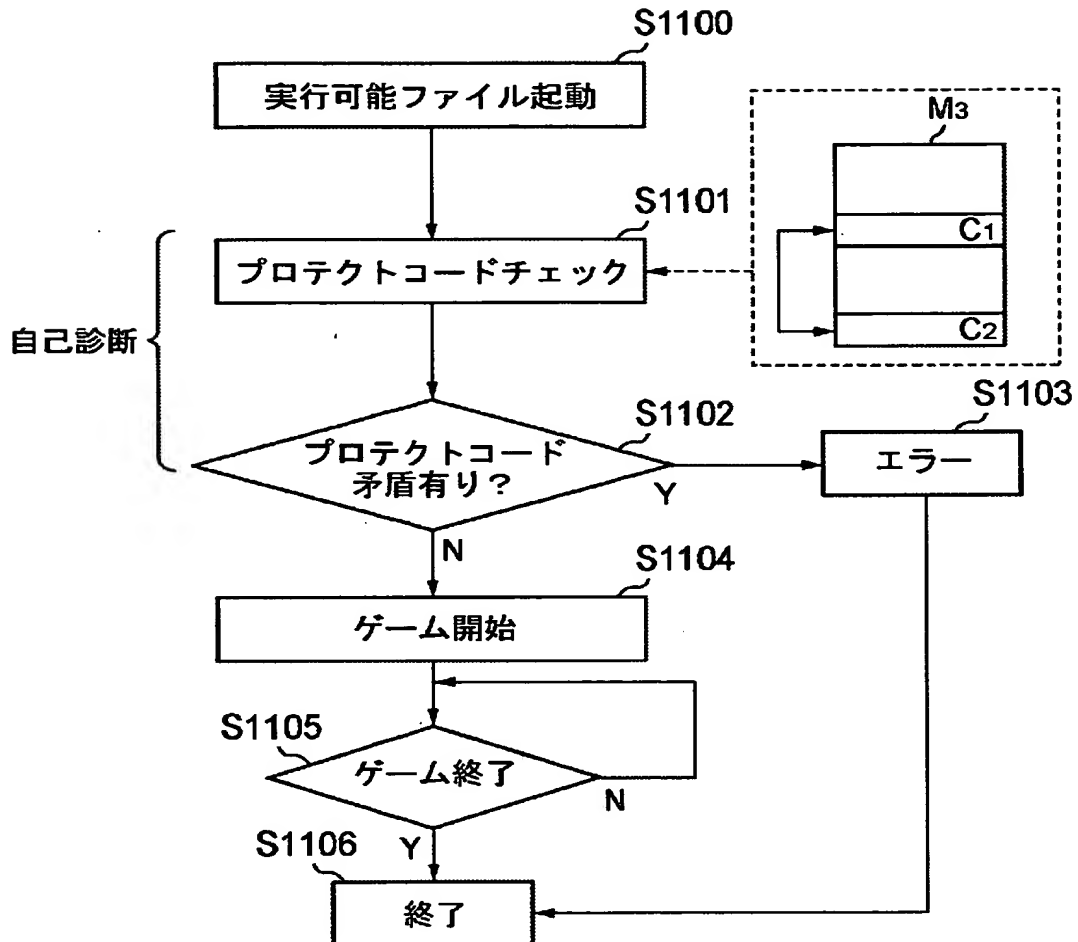
【図 1 0】

図 10



【図 1 1】

図 1 1



## 【書類名】 要約書

## 【要約】

【課題】 ライブラリ等の不正コピーを防止する。

【解決手段】 暗号化ライブラリファイル $L_2$ には、実行可能プログラムファイルに埋め込まれた2つのプロテクトコードに矛盾があれば、初期化段階で処理を終了させるプロテクトコードチェック手続きが含まれている。実行可能プログラムファイル作成時、暗号化ライブラリファイル $L_2$ が復号され、それに第一プロテクトコード $C_1$ が埋め込まれる(S 3 0 4)。そのあと、復号後のライブラリファイル $L_2$ とオブジェクトファイル $M_2$ とが結合編集され、実行可能プログラムファイルが作成される(S 3 0 5)。そして、使用済みライブラリファイル $L_2$ を消去してから(S 3 0 6)、実行可能プログラムファイル $M_3$ に、第一プロテクトコード $C_1$ と所定の関係にある第二プロテクトコード $C_2$ がさらに埋め込まれる(S 3 0 7)。

【選択図】 図 5

出 願 人 履 歴 情 報

識別番号 [395015319]

1. 変更年月日 1997年 3月31日

[変更理由] 住所変更

住 所 東京都港区赤坂7-1-1

氏 名 株式会社ソニー・コンピュータエンタテインメント